Improved Reliability of FPGA-based PUF Identification Generator Design

Gu, C., Hanley, N., & O'Neill, M. (2017). Improved Reliability of FPGA-based PUF Identification Generator Design. *ACM Transactions on Reconfigurable Technology and Systems*, *10*(3), [20]. https://doi.org/10.1145/3053681

Published in:
ACM Transactions on Reconfigurable Technology and Systems

Document Version:
Peer reviewed version

Improved Reliability of FPGA-based PUF Identification Generator Design

Chongyan Gu, CSIT, ECIT, Queen's University Belfast, Queen's Road, United Kingdom, BT3 9DT
Neil Hanley, CSIT, ECIT, Queen's University Belfast, Queen's Road, United Kingdom, BT3 9DT
Máire O'Neill, CSIT, ECIT, Queen's University Belfast, Queen's Road, United Kingdom, BT3 9DT

Physical unclonable functions (PUFs), a new form of physical security primitive, enable digital identifiers to be extracted from devices, such as field programmable gate arrays (FPGAs). Many PUF implementations have been proposed to generate these unique n-bit binary strings. However, they often offer insufficient uniqueness and reliability when implemented on FPGAs, and can consume excessive resources. To address these problems, in this paper we present an efficient, lightweight and scalable PUF identification (ID) generator circuit that offers a compact design with good uniqueness and reliability properties, and is specifically designed for FPGAs. A novel post-characterisation methodology is also proposed, which improves the reliability of a PUF without the need for any additional hardware resources. Moreover, the proposed post-characterisation method can be generally used for any FPGA-based PUF designs. The PUF ID generator consumes 8.95% of the hardware resources of a low-cost Xilinx Spartan-6 LX9 FPGA and 0.81% of a Xilinx Artix-7 FPGA. Experimental results show good uniqueness, reliability, and uniformity with no occurrence of bit-aliasing. In particular, the reliability of the PUF is close to 100% over an environmental temperature range of 25^oC to 70^oC with $\pm$10% variation in the supply voltage.

Additional Key Words and Phrases: Physical unclonable functions, identification generation, authentication, field programmable gate arrays (FPGAs), reliability.

1. INTRODUCTION

Physical unclonable functions (PUFs) allow hardware devices to be physically and uniquely identified by associating unique n-bit binary string identifiers with the devices, such as FPGAs. This has opened the door to a number of new security-orientated FPGA design opportunities, such as intellectual property protection, cloning prevention or complex security-on-chip designs. PUF designs have been extensively studied for hardware targets, both application specific ICs (ASICs) and field programmable field arrays (FPGAs), and recently have begun to appear in commercial products such as the latest version of the NXP SmartMX micro-controller integrated circuit (IC) targeted at transport and banking markets [Intrisic-ID 2015], and the new Microsemi SmartFusion2 SoC FPGA line [Microsemi 2015]. The Xilinx UltraScale+ devices also have the option of using PUF for masking of the key used for bitstream encryption (note that it is not accessible for user designs), and have published whitepapers on how to enhance boot security with the use of a user designed "soft" PUF in the FPGA fabric [Peterson 2015]. Ideally, PUFs should offer a tamper-evident, unpredictable and unclonable solution. However, due to machine learning attacks [Rührmair et al. 2010] and physical attacks, e.g. fault injection [Delvaux and Verbauwhede 2014], side channel attacks [Mahmoud et al. 2013] for some PUF designs their responses can be predicted. Recently, countermeasures have also been proposed to improve the physical security of PUF designs, e.g. modeling attack resistant PUFs [Kumar and Burleson 2014] [Vijayakumar and Kundu 2015]. Some of these proposals will be discussed in the next section. One advantage of using PUF over other approaches is that any device tampering can affect the PUF response and hence might be detected. Instead of storing a preset identifier in non-volatile memory (NVM) they exploit process variation effects via a "variability aware" circuit to generate a unique n-bit binary string identifier (response) from an FPGA when given a corresponding N-bit input (challenge). Since, ideally, each i-th challenge, C_i, uniquely maps to exactly one response, R_i, with-

out noise, they can be grouped together into so-called challenge-response pairs (CRPs), where $CRP_i = \{C_i, R_i\}$.

To enable practical use of the identifiers generated using the FPGA-based PUFs they have to 1) be unique so that no two devices map to the same ID, 2) offer high reliability to ensure that a device can repeatedly return the correct ID with as few noisy bits as possible, and 3) be efficient and feasible to implement on an FPGA. However, to date wide spread adoption in FPGAs has been limited as PUFs are difficult to implement and integrate on such devices, can require considerable FPGA logic resources, and have insufficient tolerance to temperature and voltage variations.

Uniqueness in PUF IDs is inherently difficult to achieve as they exploit manufacturing process variations, which FPGA microelectronic designers strive to minimize. If the level of variability is not sufficient, two challenges, C_1 and C_2, may map to the same or related responses for different FPGA instances i.e., $C_1 \rightarrow R_1$ and $C_2 \rightarrow R_1$. Furthermore, as FPGAs have highly regular and scalable architectures to allow them to implement arbitrary logic functions efficiently, this directly affects the implementation options. The design tools create layouts with unbalanced routing and interconnects with large capacitance, both of which introduce bit biases and skew in PUF response bits i.e., bit-aliasing.

Reliability in PUF ID generator designs is affected by environmental variations, the most significant of which are core supply voltage and temperature fluctuations. If an FPGA's core supply voltage levels diverge significantly from the recommended value gate delays will change and can cause incorrect ID responses. Similarly, elevated local temperatures will impact an FPGA's performance and response bit accuracy. Changes in temperature cause transistor threshold voltages to decrease and carrier mobility to increase: the former tends to speed up a circuit, while the latter tends to slow it down. Depending on which effect is dominant, a circuit may show either negative temperature dependence if the delay increases with temperature, positive temperature dependence if it decreases with temperature, or mixed temperature dependence if the trend is non-uniform [Wolpert and Ampadu 2012]. Ultimately, both types of environmental variation causes reliability issues.

In previous work by the authors [Gu et al. 2014], a novel FPGA-based PUF ID generator was proposed, which consumes minimal FPGA logic resources, and can be easily scaled to form an n-bit PUF ID generator and implemented in a low-cost FPGA device, such as a Spartan-6 LX9. Experimental results demonstrate that this PUF ID generator design achieves good uniqueness and reliability. Each 1-bit ID cell is implemented as a hard-macro on an FPGA ensuring balanced and stable routing for reliable operation. This is important when generating identifiers under different environmental conditions and minimizes statistical bias. Although the reliability of our previous work (93%) is sufficient for ID generation and authentication, for other applications, such as key generation, a more robust response is required. Ideally the aim is to improve the reliability result without utilising extra hardware resource on an FPGA, which is one focus of this paper.

In this paper, we build upon our previous work and propose a reliable characterisation process. We also provide a more complete analysis of the FPGA-based PUF ID generator. Specifically, our scientific research contributions are as follows:

— We propose a reliable and efficient post-processing characterisation process, which can be implemented on FPGA without any additional hardware resources. This characterisation process can be utilized to improve the reliability of any FPGA-based PUF ID generator design. It is employed to enhance the reliability of the 128-bit PUF ID generator previously proposed by the authors [Gu et al. 2014].

— The application of this automated characterisation process is presented, and an improvement in the reliability of the 128-bit PUF ID generator from 93.93% to 98.74% without the requirement of any additional hardware resources, and to 99.60% when simple majority voting post-processing is also employed, is shown.
— A more comprehensive evaluation of the PUF ID generator design previously proposed by the authors [Gu et al. 2014] is also presented, and includes an analysis of uniformity and bit-aliasing, with results of 51.06% and 56.48% achieved respectively pre-characterisation.
— The proposed improved 128-bit PUF ID generator is shown to achieve good overall results in terms of uniqueness, uniformity and bit-aliasing, with values of 45.60%, 50.60%, and 56.48% respectively using the proposed characterisation process, and a further improvement to 45.60%, 50.54% and 56.58% respectively using majority voting.

The rest of this paper is organized as follows. Section II discusses the related work of PUF designs. Section III introduces the principle of the PUF ID generator's operation. The implementation of the 128-bit PUF ID generator design is described in section IV and the post-characterisation process is outlined in section V. The evaluation of the proposed improved PUF ID generator design is given in section VI to validate the work. Finally, conclusions are drawn in section VII.

2. RELATED WORK

Since the first concrete implementation of a PUF was proposed [Pappu et al. 2002], many researchers have reported a range of different PUFs targeting both ASICs and FPGAs, e.g. static RAM (SRAM) PUFs [Guajardo et al. 2007; Holcomb et al. 2009], Latch PUF [Su et al. 2008], Flip-flop PUF [Maes et al. 2008], Buskeeper PUF [Simons et al. 2012], Butterfly PUF [Kumar et al. 2008], Ring Oscillator (RO) PUF [Suh and Devadas 2007; Murphy et al. 2012], Configurable RO (CRO) PUF [Yu et al. 2012], Arbiter PUF [Gassend et al. 2002] [Gu et al. 2016], Bistable Ring (BR) PUF [Chen et al. 2011], processor-based PUF [Maiti and Schaumont 2012], reconfigurable PUF (rPUF) [Kursawe et al. 2009]. Also, Charles *et al.* [Herder et al. 2014] and Sklavos [Sklavos 2013] provide a detailed introduction to PUF based security analysis and implementation. The RO PUF designs exploit the difference in period between two identical ring oscillators by incrementing two counters and then comparing the value reached in a given time frame. This structure at a minimum requires two configurable logic blocks (CLBs) on FPGA, even though strategies exist to re-use oscillators [Maiti et al. 2012]. The Arbiter PUF uses n-bit differential delay lines and a latch arbiter to generate a 1-bit PUF response. It is difficult to implement this design on FPGA as it requires the whole structure to be balanced to generate 1-bit, and then duplicated k times. Although Majzoobi *et al.* [Majzoobi et al. 2014] and Hori *et al.* [Hori et al. 2014] implemented Arbiter PUFs on FPGAs, they introduced an extra tuning circuit or reported results with low uniqueness. In memory-based PUFs, like SRAM PUFs, the initial state of static RAM cells, formed by two cross coupled inverters (also often known as a bistable latch) in FPGAs, is exploited to produce IDs based on different memory blocks on different FPGAs. However, SRAM PUFs require a device power-up operation to generate each ID. Although most FPGAs have SRAM memory, some SRAMs have an initial state which prevents them entering a random value during the start up stage. To address this Kumar *et al.* [Kumar et al. 2008] proposed a logical alternative called Butterfly PUF to emulate the behavior of an SRAM PUF on Virtex-5 FPGAs. It can be invoked at any time rather than only at power-up. It operates using two cross-coupled latches forming a bistable circuit, where the preset/clear force it into metastability. It still suffers from issues due to metastability, and indeed not all

FPGAs feature preset/reset pins in the required format. They reported 94% reliability over temperature variations, however, reliability over voltage changes is not provided. For 64 Butterfly PUF cells, 130 slices are consumed.

Improving the reliability has been the subject of much research with both SRAM and RO PUF designs. Efforts to improve the reliability of SRAM PUF have been proposed by many researchers, e.g. Bhargava *et al.* [Bhargava and Mai 2014], Garg *et al.* [Gary and Kim 2014] and Cortez *et al.* [Cortez et al. 2013]. However, aging testing based reinforcement techniques or special circuitry are required. Guajardo *et al.* [Guajardo et al. 2007] and Bohm *et al.* [Bohm et al. 2011] utilise error correction codes, BCH code or repetition code, to reduce the error rate for SRAM PUFs. A fuzzy extractor is used for error correction to enhance the reliability in the SRAM PUF design proposed by Holcomb *et al.* [Holcomb et al. 2013], the Flip-flop PUF design by Maes *et al.* [Maes et al. 2008], and the Butterfly PUF design by Kumar *et al.* [Kumar et al. 2008]. All these post processing methods incur additional hardware resource usage. Majority voting (or 1-out-of-k-method) is a straightforward and simple way to reduce noise. However, its usefulness depends on the level of noise. For example, in the case of 50% noise, even if majority voting is applied, the result cannot be improved. Hence, a lightweight reliability improvement is needed to enhance the reliability of FPGA-based PUF designs in general.

To distinguish between different intrinsic PUF designs, Guajardo *et al.* [Guajardo et al. 2007] introduced two PUF subtypes with regard to the behavior of CRPs, namely "Weak PUF" and "Strong PUF". Weak PUFs exhibit the following characteristics: 1) they may have very few challenges and in the extreme case they may generate just one response; 2) it is assumed that an attacker can not access the response of Weak PUFs as one or a few CRPs could be used to build a model of the security system. The previously mentioned SRAM PUF and Butterfly PUF are examples of Weak PUFs. For many applications, their responses can be useful for key generation as an intrinsic key, in place of secure memory. Compared to other key storage approaches in which keys are stored in NVM, the key is intrinsically linked to the physical hardware of the device itself. Furthermore, Weak PUFs are low-cost since they do not need any special manufacturing process. Compared to Weak PUFs, Strong PUFs have the following characteristics: 1) they may have many possible CRPs; 2) an attacker may have access to the CRPs, however it should be impossible for them to determine or attack the CRPs in a given time frame, for example, a few days or weeks.

Practical realisations of the Strong PUF definition of Guajardo *et al.* [Guajardo et al. 2007] has proven to be somewhat more difficult than originally anticipated. For example, the Arbiter PUF was an example of a Strong PUF, however, it is known that Arbiter PUFs can be modeled as linear additive models [Rührmair et al. 2010], and recent work has shown how to attack the non-linearity of XOR-Arbiter PUFs using reliability-based evolution strategies [Becker 2015]. Hence, in more recent literature, a Strong PUF is required to have a number of CRPs that scales exponentially with the circuit area, but no other constraints such as resistance against modeling are imposed. For Weak PUF, such attacks do not apply as it is assumed that there is no external access to the response for an attacker. Hence, machine learning attacks are not considered here. For Strong PUFs, poor reliability is one reason why XOR PUFs can be attacked, as described in [Becker 2015]. Hence, the characterisation process presented in this paper to help achieve high reliability, may be helpful for Strong PUFs to protect against machine learning attacks.

3. PUF ID GENERATOR CIRCUIT DESIGN

The previously proposed PUF ID generator design by the authors [Gu et al. 2014] comprises of n elementary 1-bit PUF ID cells and is designed to fit compactly in one FPGA

slice, as shown in Fig.1. An n-bit PUF ID generator circuit is formed by instantiating an array of n 1-bit ID cells. A 1-bit response is generated as follows: two matched time delay paths, *T0* and *T1*, implemented by two D type flip-flops are excited simultaneously by the rising edge of a *START* signal connected to their clock pins after first being reset by *CLEAR*; since flip-flops are coarse grained delay components, the rising edge on their *Q* outputs propagate the excited signal differently, thus racing against each other; the timing of two delay lines will differ due to underlying manufacturing variability; cross-coupled NAND gates are utilized to decide which transition arrived first and sets their output to either binary 10 or 01. A timing diagram of the 1-bit PUF ID generator design is shown in Fig. 2. It can be seen that a *CLEAR* signal is first activated to reset the circuit and on the rising edge of a *START* signal the delay paths are activated. The output signals, *Z0* and *Z1*, will be 01 when *Q0* and *Q1* are 10 whenever the arrival time of the delay path *T0* is faster. A multiplexer outputs a unique 1-bit response depending on the value of the challenge. This PUF ID generator bit generation circuitry requires two LUTs, two flip-flops and one multiplexer per bit.

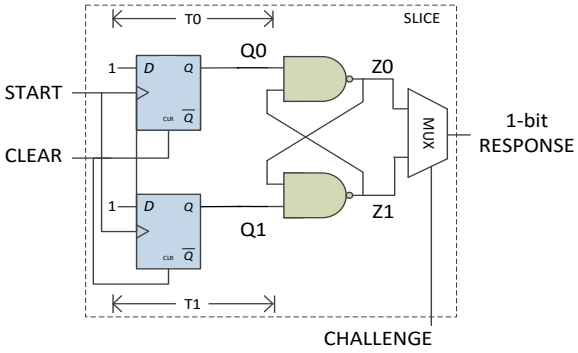


Fig. 1. circuit design.

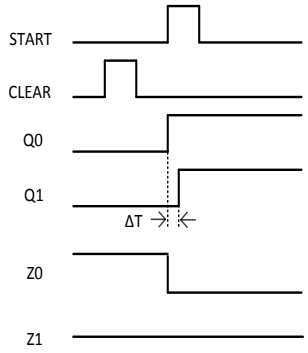


Fig. 2. Timing diagram.

The use of cross-coupled NAND gates as an arbiter ensures that the feedback paths are balanced, symmetrical and contribute minimally to *T0* and *T1*. The arbiter design also increases reliability as the effects on each feedback path are equally balanced. Previous work by Lim *et al.* [Lim et al. 2005] uses a D-latch for the arbiter, but it introduces a 10% skew on the response.

In order to maximize variation and to avoid bit aliasing in the ID responses the wiring paths must be placed and routed as symmetrically as possible so as to minimize the nominal delay difference between the two paths. In FPGAs this can be accomplished by manual routing and timing analysis, but due to the natural architecture of FPGAs this is inherently problematic. Careful place and route in the target device ensures an estimated delay difference between the paths of only 10 ps according to the design tool.

Due to external influences as mentioned previously (temperature and supply voltage), some bits will be unstable and vary between 0 and 1. The straight forward solution to this problem is to obtain each response bit N (N=5 for this work) times and then use the majority as the correct bit. As the number of repetitions increases, the probability of an undecipherable error decreases proportionately. This does not work however when significant instability in the bit response occurs. In the next section, a characterisation process is proposed to address instability and improve reliability.

4. IMPLEMENTATION OF PROPOSED PUF ID GENERATOR DESIGN

The previously proposed n-bit PUF ID generator design is implemented in Xilinx Spartan-6 FPGA and each bit is implemented as a hard macro, as shown in Fig.3. The floor plan location is set by declaring location (LOC) constraints using Xilinx's Unified Constraints Format (UCF) file.

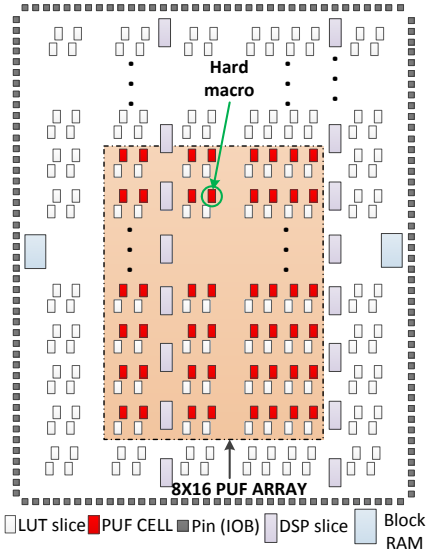


Fig. 3. Floor plan of a 128-bit PUF ID generator based on a 1-bit single slice hard macro.

To ensure that the ID response bits are a function of device variability only, it is essential that all circuit elements are identical and routing is balanced. Otherwise, the response bits will exhibit bit-bias due to the interconnect and layout mismatch. The authors in [Suh and Devadas 2007] suggest using FPGA hard-macros as a solution to help meet strict design parameters. Therefore, in this research the circuit elements are manually placed and routed and a hard-macro, as shown in Fig.4, is used to implement a 1-bit ID cell, which occupies exactly one slice of the target Xilinx Spartan-6 FPGA device. Fig.5 shows an example of the unbalanced routing that results from automated place and route of the design, which will lead to a bias of the response. Other FPGA families can be targeted using the 1-bit ID cell by re-creating the hard-macro. A 5-bit majority voting circuit for each 1-bit ID cell is also implemented as a hard macro in one slice.

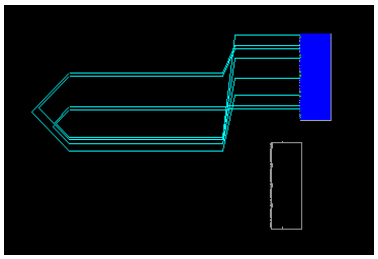


Fig. 4. Balanced routing

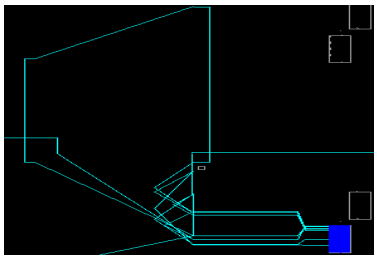


Fig. 5. unbalanced routing

The cross-coupled NAND gate arbiter is implemented in two LUTs, the D type flip flops are implemented in two registers; and the 2:1 selector is implemented in a multiplexer. These are easily instantiated in hardware description language (HDL) (e.g., Verilog) by using a stub file, which can be declared multiple times as desired to build n-bit PUF ID generators. In this work 128 hard-macros, arranged in an 8×16 array, are used to construct the 128-bit PUF ID generator as shown in Fig. 3. The Xilinx Spartan-6 device employed in this work is the XC6SLX9, which has 1,430 slices, half of which are SLICEXs, a quarter of which are SLICELs and a quarter SLICEMs [Xilinx 2011]. Compared to SLICEX, the SLICEL and SLICEM primitives have wide MUXs and carry chain components. As the proposed design does not need wide MUXs or carry chains it can be implemented in any type of slice, and for this work it is implemented in SLICEX primitives.

Since a 1-bit ID cell only occupies half a Spartan-6 slice, the remaining resources can be used to implement other functionality or alternatively a second 1-bit ID cell yielding 2-bits of an ID response per slice [Gu and O'Neill 2015]. For this work, a single bit per slice is implemented, such that each ID cell occupies the upper half of the slice and the upper slice within a CLB. Utilizing only half a slice allows a great amount of flexibility in the design of complex systems as the ID hard-macro cells can be placed anywhere on the FPGA floor plan to maximize overall resource consumption and to minimize routing congestion.

5. POST-CHARACTERISATION METHODOLOGY

To improve the reliability of the PUF ID generator design, a post-characterisation phase to analyse the robustness is introduced to find the unstable bits in an FPGA layout. Robustness represents how reliable the PUF ID generator design is at nominal supply voltage and room temperature. It is generally calculated by the intra-chip hamming distance between S sample responses from the same PUF device under the same operational conditions. A manual characterisation process is employed to prove the feasibility of this methodology.

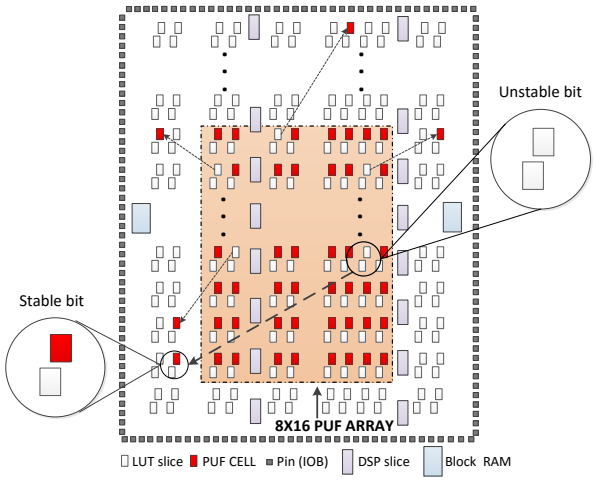


Fig. 6. Characterising the floor plan of a 128-bit PUF ID generator based on a 1-bit single slice hard macro.

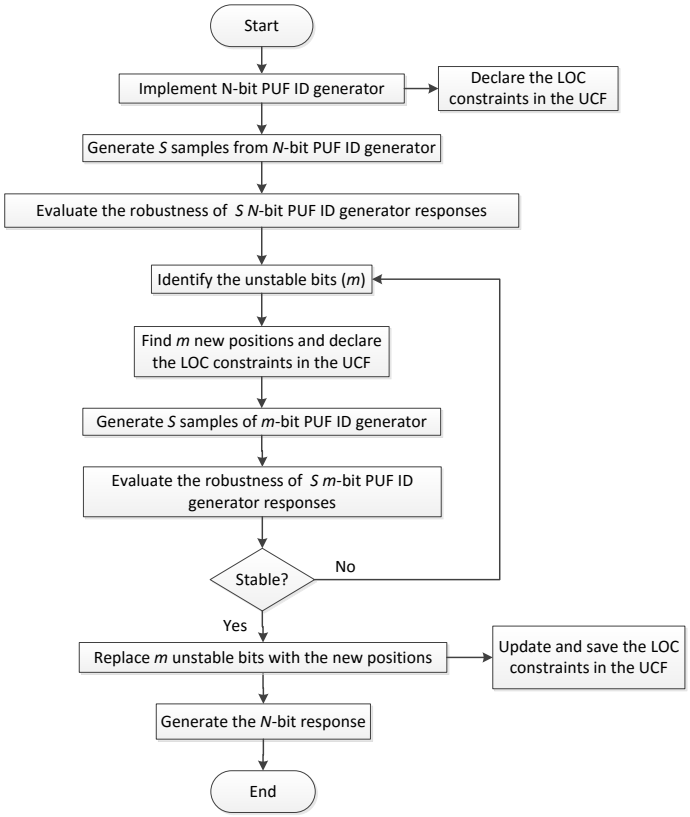


Fig. 7. The flow chart of characterising the floor plan of PUF ID generator IP core architecture.

5.1. Manual Characterisation Process

A flow chart outlining the steps involved in the characterisation process is shown in Fig.7. The steps are as follows:

— Implement an N-bit PUF ID generator in the target FPGA, e.g. Spartan-6, as shown in Fig.3.
— Generate S responses from the N-bit PUF ID generator, where S is the sample number
— Evaluate the robustness of S N-bit PUF ID generator responses (robustness represents the reliability of the PUF design under normal operating conditions);
— Identify the unstable bits (m) and their positions in the floor plan of the FPGA,
— Find m stable bits and their position as shown in Fig.6;
— Move the PUF cells in unstable bit positions to stable bit positions (repeat several times until all PUF cells positions are stable). The placement of the PUF ID cells is achieved by manually declaring LOC constraints using Xilinx's UCF in the Xilinx ISE tool;
— Update and save the final bit file with the new bit positions as the default floor plan;
— Generate the N-bit PUF ID generator response.

In this work the characterisation process was first manually executed following the above steps, which demonstrates the feasibility of the technique. However, an automated post-processing characterisation process was also considered to improve the efficiency of the approach.

5.2. Automated Characterisation Process

The flow chart for the automated characterisation process is similar to that of the manual process and is shown in Fig.8. Algorithm 1 describes in detail the execution of each step in the process and includes six phases. Similar to the manual characterisation process, the automated process explores the most stable bit output for the response in order to improve the reliability of the PUF ID generator design. Moreover, the automated characterisation process simplifies the post processing, and only needs to be carried out once to find out all the unstable and stable bit positions over the whole FPGA device. The execution time depends on the size of the FPGA, where the larger the FPGA, the longer the time it takes to identify the unstable and stable bit positions.

ALGORITHM 1: Pseudo-code for automated characterisation process

1: **Phase0: Setup**
2: Declare the position of an M -bit PUF ID cell in the UCF, where M is the maximum available slices of the target FPGA
3: Implement M -bit PUF ID generator design on the FPGA
4: **Phase1: PUF Response**
5: **for** $i = 0$ to S **do** (where S is the sample number for robustness)
6: Generate the response of the M -bit PUF ID generator
7: **end for**
8: **Phase2: Evaluate**
9: Evaluate the robustness of S M -bit PUF ID generator responses
10: **Phase3: Identify**
11: Identify the unstable bits (p) and the stable bits (q) of the robustness result
12: Note the position of the stable bits (q)
13: **Phase4: Choose**
14: Randomly choose the position of n stable bits from the q bits as the position of N -bit PUF ID generator, where N is the required bit length of the PUF ID generator
15: Update and declare the position of the N -bit PUF ID cell in the UCF
16: **Phase5: Generate**
17: Generate the response of the N -bit PUF ID generator

6. EVALUATION OF PROPOSED IMPROVED PUF ID GENERATOR DESIGN

6.1. Experimental Setup

The Xilinx ISE Design Suite 14.7 tool was used for the proposed design and Matlab was utilised to communicate with and test the PUF IP core, with a simple interface written to send and receive data over the USB-UART port of target FPGA boards. To evaluate the manual characterisation process, the 128-bit PUF ID generator design was implemented on a Spartan XC6SLX9 microboard which comprises of a low-cost Xilinx Spartan-6 (CSG324) FPGA device (45nm technology). The 128-bit identification generator design was programmed into ten identical Spartan-6 LX9 Microboards as shown in Fig.9. One was manually modified to conduct temperature and voltage experiments by varying the core voltage ($\pm 10\%$) and the temperature from 25^oC to 70^oC as shown in Fig. 9.

To evaluate the automated characterisation process, the 128-bit PUF ID design was implemented on Digilent Nexys4 microboard which comprises of a Xilinx Artix-7

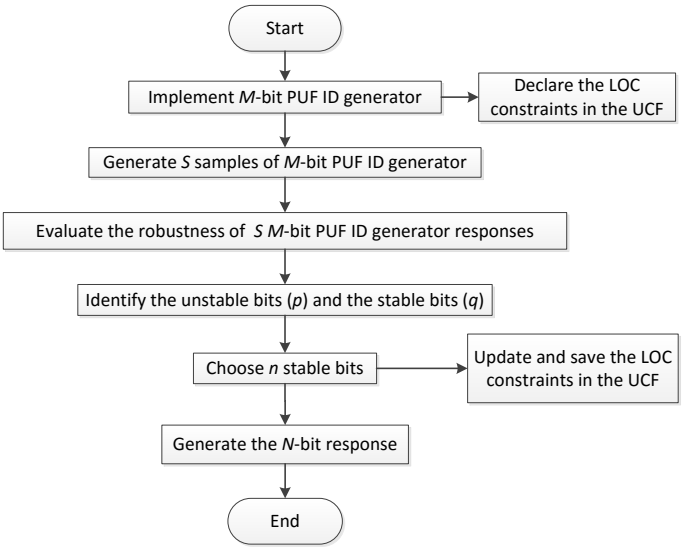
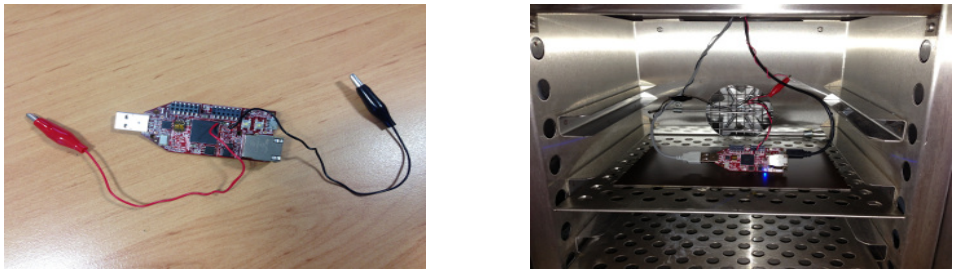


Fig. 8. Flow chart of automated characterisation process



Modified LX9 microboard LX9 microboard in heat chamber

Fig. 9. Experimental setup

XC7A100T FPGA (28nm technology) to prove its feasibility on a more recent technology. The communication and control units on the Xilinx Artix-7 FPGA are similar to those on the Xilinx Spartan-6. One Xilinx Artix-7 FPGA was again modified to conduct temperature and voltage experiments. The core voltage was varied by $\pm 10\%$ and the temperature from 0^oC to 75^oC. The core voltage of the Artix-7 FPGA is 1.0 volts, which differs from the Spartan-6 FPGA which has a core voltage of 1.2 volts.

There are four important metrics used to quantify the performance of a PUF ID generator circuit: uniqueness, reliability, uniformity and bit-aliasing. These are used to evaluate the improved PUF ID generator design.

6.2. Uniqueness

Uniqueness measures inter-chip variation by evaluating how easily a particular PUF ID generator circuit design can differentiate between k different devices. Specifically, it quantifies the average inter-chip hamming distances (HDs) between sets of responses extracted from different devices, which implement the same PUF ID generator circuit

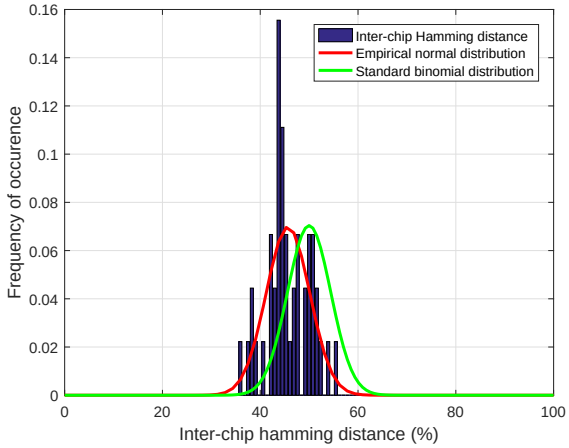


Fig. 10. Uniqueness result

and have been supplied with the same challenge, to show the extent of the difference of responses.

Ideally, when a PUF ID generator circuit is implemented on different devices it should produce an average inter-chip HD of 50% when compared between two devices supplied with the same challenge, implying that, on average, half the response bits are different between the two devices even though the same challenge has been used.

Accordingly, a percentage figure-of-merit for uniqueness based on average inter-chip HD can be defined. If two chips i and j both implement the same PUF ID generator circuit and have n-bit responses R_i and R_j to the same challenge, C, then uniqueness expressed as the average inter-chip HD among k devices is defined as:

$$\text{Uniqueness} = \frac{2}{k(k-1)} \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} \frac{\text{HD}(R_i, R_j)}{N} \times 100 \tag{1}$$

A probability density function (PDF) plot of the $\frac{\text{HD}(R_i,R_j)}{N} \times 100$ values is shown in Fig.10, where values near to 50% indicate higher uniqueness. The distribution is Gaussian, clustering around the uniqueness value of 45.60%. This confirms a uniqueness approaching the ideal value can be expected for the proposed PUF ID generator on this particular FPGA.

6.3. Reliability

Ideally a given PUF ID generator circuit, implemented in any device should be able to perfectly reproduce its output whenever it is queried with a challenge. However in practice, environmental changes, such as temperature and power supply voltage variations, as well as the natural properties of metastability in PUF ID generator circuits induce noise in the responses. Therefore, reliability is used to quantify a PUF ID generator's ability to reproduce a response. Reliability can be regarded as a percentage measure of the number of noisy ID response bits.

For a device i, reliability is established as a single value by finding the average intra-chip HD of s response samples, $R_i^{'}$, taken at different operating conditions compared

to a baseline N-bit reference response, R_i, taken at nominal operating conditions. The average intra-chip HD is estimated as follows:

$$\mathsf{HD}_{\mathsf{INTRA}} = \frac{1}{s}\sum_{t=1}^{s}\frac{\mathsf{HD}(R_i, R'_{i,t})}{N} \times 100 \tag{2}$$

where $R(i,t)^{'}$ is the $t-$th sample of R'_i. The percentage figure of merit for reliability can be defined as:

$$\mathsf{Reliability} = 100 - \mathsf{HD}_{\mathsf{INTRA}} \tag{3}$$

Obviously, the ideal value for reliability is 100%. To investigate the reliability of the proposed PUF ID generator design, a 128-bit reference response R_i was extracted from a chip i under normal conditions, that is at room temperature and with normal supply voltage. This is compared with the responses R'_i, taken under varying operating conditions. Note, robustness can be calculated using the same formula as reliability. The only difference is that the response samples for robustness are derived under nominal operating conditions.

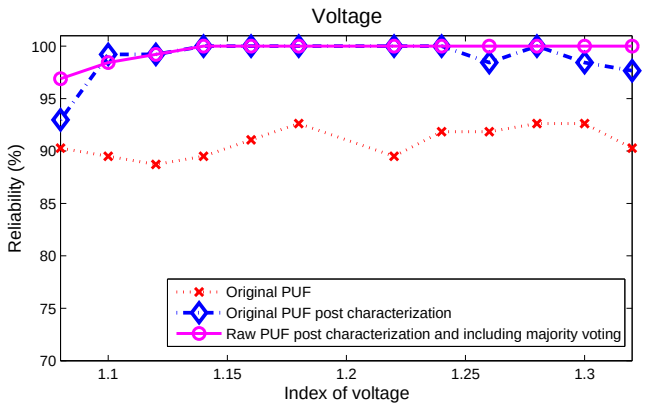


Fig. 11. Reliability comparison over supply voltage variation

6.3.1. Reliability with manual characterisation process. The temperature was varied from $25\,^oC$ to $70\,^oC$ ($5\,^oC$ each step) using a convection heat chamber while the core supply voltage was varied by $\pm 10\%$ 0.2 Volts using a DC regulated power supply. This covered the permitted operating range of the FPGA and swept all combinations of operating points. Fig.11 and Fig.12 show the results and compare the responses from the previously presented PUF ID generator design, with those from the PUF ID generator design post-characterisation, as well as the PUF ID generator design post-characterisation that includes majority function circuit. Over all voltage operating points the improved PUF ID generator designs exhibit a high level of reliability of between 93% and 100% for the design post-characterisation, and between 96.5% and 100% for the design that employs both characterisation and majority function circuitry. For the temperature results, both improved PUF ID generator designs achieve reliability of 100% for the FPGAs under evaluation. As expected, the designs that use the characterisation and majority function circuitry are more reliable than the original

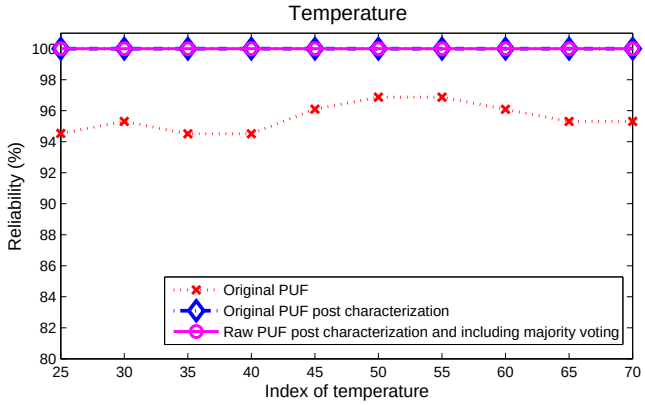


Fig. 12. Reliability comparison over ambient temperature variation

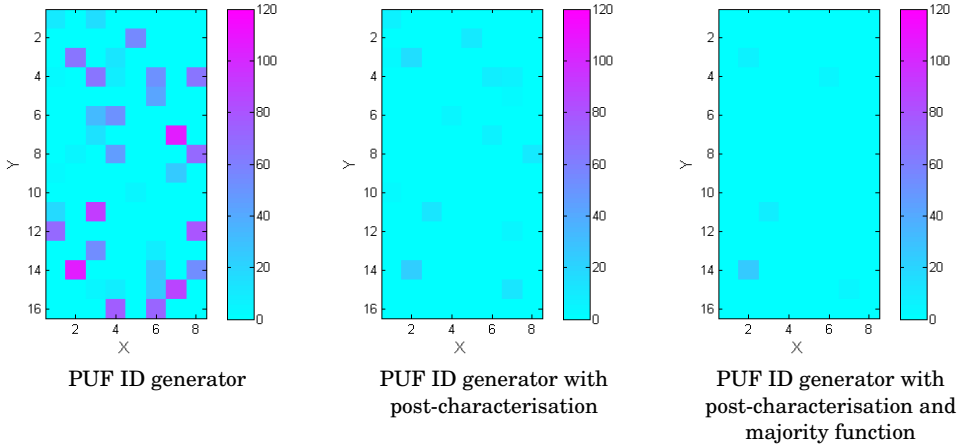


Fig. 13. Reliability variation maps of 128-bit responses based on the PUF ID generator design, the PUF ID generator design post-characterisation and the PUF ID generator design post-characterisation with majority function circuitry.

PUF ID generator design. The average reliability results are listed in Table V. As discussed in the previous section the improvement is due to changing the position of the unreliable 1-bit ID cells which improves the robustness of the response bits.

To estimate the fluctuation in reliability of each bit, the variation of each response bit was compared under normal operating conditions and varying operating conditions. A variation map is used to explain the effects of the ID cell positioning on the response bits. A group of 128-bit responses was extracted under various environmental conditions. The variation from the original response bit is obtained by comparing these responses to the reference response obtained under normal conditions. At each specific position, the difference between the derived response and the reference response can be represented as either *'0'* or *'1'*, where *'0'* means no difference between responses and *'1'* means the response from the conditioned situation is different to the one obtained under normal conditions. Therefore, the sum of the difference in responses at each position $S_{r_{i,l}}$ can be found, where $r_{i,l}$ denotes the l-th position bit on the i-th chip. Fig.13 shows the distribution of differences ($S_{r_{i,l}}$) from the reliability results for the 128-bit

positions. X represents the bit position on the x-axis of the floor plan in Fig.3, and Y represents the bit position on the y-axis of the floor plan. The ranges of X and Y are $1 \rightarrow 8$ and $1 \rightarrow 16$ respectively, which indicate the response position as follows:

$$r_{i,l} \Rightarrow \{l_X, l_Y\} \tag{4}$$

where l_X and l_Y are the x-axis and y-axis response position $r_{i,l}$. The subfigures respectively show the distribution of the difference on reliability for the PUF ID generator design, the PUF ID generator design using the characterisation process and the PUF ID generator design post-characterisation and including the majority function circuitry. The colorbar ranges are $0 \rightarrow 120$ in Fig.13, where the samples are from 12 different voltages$\times 10$ different temperatures. The darker color indicates that the difference value, $S_{r_{i,l}}$, at position $r_{i,l}$ is larger. In other words, the PUF ID generator exhibits more instability at specific positions. After the manual characterisation process, this instability is reduced. Moreover, using the majority function, the variation of the response is essentially eliminated. It is clear that the characterisation process significantly improves the PUF ID generator design in terms of reliability, and by itself provides close to optimal results.

Table I. BER results for voltage variations for the original design, the design post-characterisation and the design post-characterisation and including error correction circuitry.

Design	BER (%)												SD	Mean
	1.08v	1.10v	1.12v	1.14v	1.16v	1.18v	1.22v	1.24v	1.26v	1.28v	1.30v	1.32v	(%)	(%)
Original	9.2	9.8	10.4	9.8	8.6	7.3	9.8	8.0	8.0	7.3	7.3	9.2	1.1	8.7
CHAR	5.5	6.1	6.1	0	0	0	0	0	1.2	0	1.2	1.8	1.6	0.9
CHAR & MAJ	2.4	1.2	0.6	0	0	0	0	0	0	0	0	0	0.8	0.4

Table II. BER results for temperature variations for the original design, the design post-characterisation and the design post-characterisation and including error correction circuitry.

Design	BER (%)										SD	Mean
	25 °C	30 °C	35 °C	40 °C	45 °C	50 °C	55 °C	60 °C	65 °C	70 °C	(%)	(%)
Original	4.3	3.7	4.3	4.3	3.1	2.5	2.5	3.1	3.7	3.7	0.7	3.5
CHAR	0	0	0	0	0	0	0	0	0	0	0	0
CHAR & MAJ	0	0	0	0	0	0	0	0	0	0	0	0

The bit error rate (BER) of the PUF responses for different voltages and temperatures was investigated. Table I and Table II show the BER, standard deviation (SD) and mean (Mean) values across the voltage range from 1.08 v to 1.32 v and across the temperature range from 25 °C to 70 °C. The PUF ID generator design post-characterisation and including majority voting provides the lowest BER, SD and Mean when the voltage is varied. Interestingly decreasing the core voltage has little effect on the BER. The $SD = 0.8\% \times 128bit = 1bit$, which means that there is variation in just 1-bit of the 128 bit response, and the average BER is 0.4%. The BER, SD and Mean values of both the design post-characterisation (CHAR) and the design post-characterisation and including error correction circuitry (CHAR & MAJ) over temperature changes are zero indicating that no errors occur in these designs.

6.3.2. Reliability with automated characterisation process. Initially the robustness of the PUF ID cell design (that is the intra HD of responses from the same device under nominal operating conditions) on an Artix-7 device was calculated for every slice to visualise the variability across a more recent FPGA.

A 128-bit reference response, R_i, is extracted from a chip i (Artix-7 FPGA) at room temperature and with normal supply voltage. This is compared to the responses R_i^s, taken under the same operating conditions, where $s = 1000$ samples. Eq.2 and Eq.3

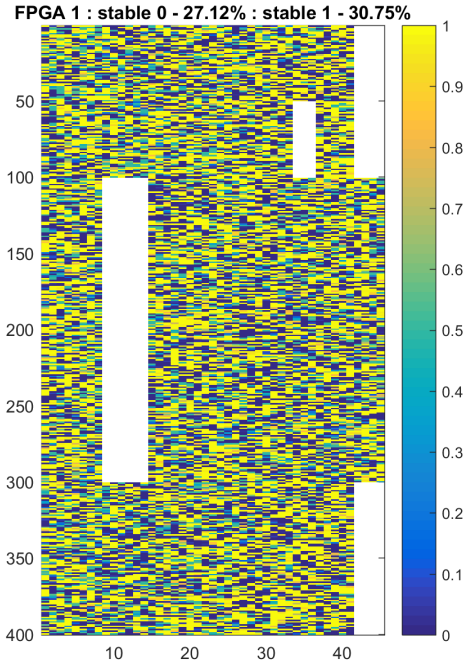


Fig. 14. Heat map of stable and unstable bits of the compact PUF ID generator design on an Artix-7 FPGA

are used in calculate robustness. Fig.14 depicts the heat map which shows the stable and unstable bits of a sample chip i. The percentage of stable '0' bits is 27.12%, the percentage of stable '1' bits is 30.75%, and the percentage of the remaining unstable bits is 42.13%. The robustness distribution of stable and unstable bits in the Artix-7 FPGA is shown in Fig.15.

Table III presents detailed robustness results of the stable '0' bits, stable '1' bits and the unstable bits on each Artix-7 FPGA, and also presents the mean and standard deviation *(STD)* values over 10 FPGAs. It can be seen that the distribution of stable '0' bits, stable '1' bits and unstable bits on the 10 FPGAs is very similar and has a very small STD.

To evaluate the reliability of the proposed PUF ID generator design when the automated characterisation process is applied, 128-bit reference responses, R_i, were extracted from a chip i under normal conditions. These are compared with responses $R_i^{'}$, taken under varying operating conditions. Temperature was varied from $0\,^oC$ to $75\,^oC$ (in steps of $5\,^oC$) using a thermal electric plate while the core supply voltage was varied by 1.0 Volts $\pm 10\%$ using a DC regulated power supply. This covered the permitted operating range of the FPGA and swept all combinations of operating points. Fig.16 shows the results and compares the responses from the PUF ID generator design, with those from the PUF ID generator design post auto-characterisation as well as the PUF ID generator design post auto-characterisation that includes majority voting. Over all voltage operating points the improved PUF ID generator designs exhibit a high level of reliability of between 95% and 100% for the design post auto-characterisation, and between 97% and 100% for the design that employs both auto-characterisation and

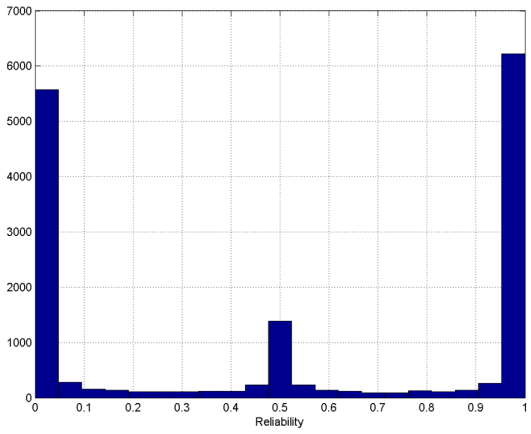


Fig. 15. Robustness distribution of the compact PUF ID generator in Artix-7 FPGA

Table III. Robustness results of stable '0' bits, stable '1' bits, and unstable bits on the ten Artix-7 FPGAs

FPGA	Bits	Stable 0's	Stable 1's	Unstable
1	15850	4299 (27.12%)	4874 (30.75%)	6677 (42.12%)
2	15850	4951 (31.23%)	4388 (27.68%)	6511 (41.07%)
3	15850	4579 (28.89%)	4728 (29.83%)	6543 (41.28%)
4	15850	4242 (26.76%)	4831 (30.47%)	6777 (42.75%)
5	15850	4986 (31.45%)	4832 (30.48%)	6032 (38.05%)
6	15850	5505 (34.73%)	5492 (34.65%)	4853 (30.61%)
7	15850	4338 (27.36%)	4582 (28.90%)	6930 (43.72%)
8	15850	4048 (25.53%)	5080 (32.05%)	6722 (42.41%)
9	15850	4262 (26.89%)	5113 (32.25%)	6475 (40.85%)
10	15850	5572 (35.15%)	4041 (25.49%)	6237 (39.35%)
Mean		4678 (29.51%)	4796 (30.25%)	6376 (40.22%)
STD		546 (3.44%)	403 (2.54%)	595 (3.75%)
All		1 (0.006%)	0 (0.000%)	15849 (99.99%)

majority voting. For the temperature results, both improved PUF ID generator designs achieve reliability close to 100%.

Table IV. Reliability results of manual and automated characterisation processes on Spartan-6 and Artix-7 FPGAs

FPGA	Over voltage variation (%)			Over temperature variation (%)			Average (%)		
Spartan6	Original	Manual CHAR	Manual CHAR+MAJ	Original	Manual CHAR	Manual CHAR+MAJ	Original	Manual CHAR	Manual CHAR+MAJ
	90.87%	98.83%	99.55%	95.54%	100%	100%	93.21%	99.42%	99.78%
Artix7	Original	Auto CHAR	Manual CHAR+MAJ	Original	Auto CHAR	Auto CHAR+MAJ	Original	Auto CHAR	Auto CHAR+MAJ
	91.40%	97.58%	99.35%	96.46%	99.89%	99.84%	93.93%	98.74%	99.60%

A comparison of the improved designs post-manual characterisation and post-automated characterisation are provided in Table IV. It is clear from the results that the post-characterisation processes are an effective way to improve the reliability of the PUF ID generator design.

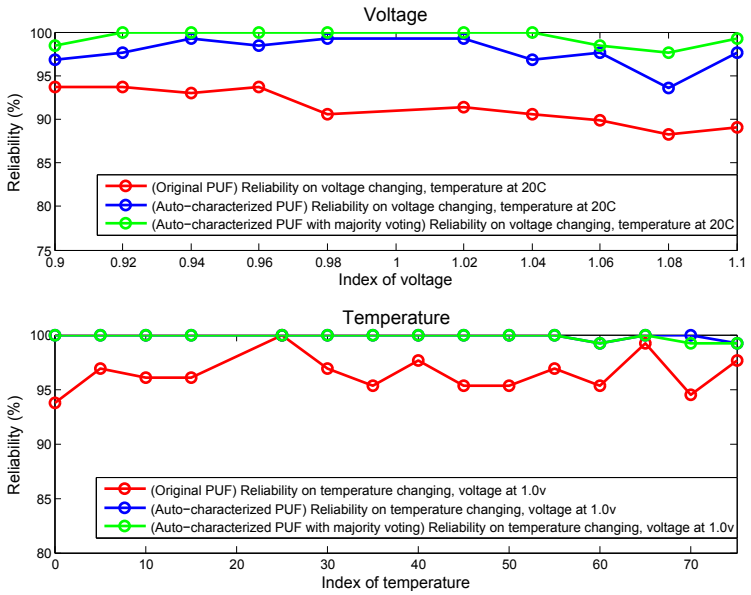


Fig. 16. Reliability results of 128-bit PUF ID generator based on the original PUF ID generator design, the PUF ID generator design post auto-characterisation and the PUF ID generator design post auto-characterisation with majority voting.

6.4. Uniformity

The uniformity of a PUF ID generator circuit measures the proportion of binary ones and zeros in a response, i.e. one-to-zero ratio, and the likelihood of each value. If a response possesses ideal uniformity and is truly random the distribution of bit values will be 50% ones and zeros. Having this property is required from a security perspective to prevent an attacker from guessing if a response of a particular device is biased towards a particular value. To estimate uniformity is simply a matter of finding the hamming weight (HW) of a response, which will reveal the proportion of ones and zeros, and any biases.

For device i and an n-bit response the percentage HW of the $n-$bit response is given as follows:

$$(\mathsf{HW})_l = \frac{1}{n}\sum_{l=1}^{n} r_{i,l} \times 100 \tag{5}$$

where, $r_{i,l}$ is the l-th position of the response bit on the i-th chip.

The proportion of 0's and 1's in a response is expected to be close to 50%, and in this work the response uniformity is 51.06% for the original PUF ID generator design, 50.60% for the PUF ID generator design post-characterisation, and 50.54% for the PUF ID generator design post-characterisation and including the majority function circuitry, as shown in Table V.

6.5. Bit Aliasing

An effective PUF ID generator design should not exhibit bit-aliasing when implemented on different devices. Bit-aliasing is when the ID response at stable positions on different chips is identical or almost identical. To determine if bit-aliasing occurs,

the total number of 0s and 1s in a response from the same $p-$th position of $k-$devices is calculated using the HW as follows:

$$(\mathsf{HW})_p = \frac{1}{k}\sum_{i=1}^{k} r_{i,p} \times 100 \tag{6}$$

where, $r_{p,i}$ is the p-th position of a response bit on the i-th chip.

If bit-aliasing occurs and different devices generate the same response from many physical positions, the security guarantees no longer hold. The percentage of 0s and 1s at the same position in ten chips $(k = 10)$ is evaluated and is shown in Table V. The value for each of the designs is 56.48%, which means that all of the bit positions are sufficiently different such that bit-aliasing is avoided for all three PUF ID generator designs.

Table V. PUF ID generator results of the original design, the design post-characterisation and the design post-characterisation and including error correction circuitry.

Metrics	Original	CHAR	CHAR & MAJ
Uniqueness	48.52%	45.60%	45.60%
Reliability	92.00%	98.97%	99.58%
Uniformity	51.06%	50.60%	50.54%
Non-bit-aliasing	56.48%	56.48%	56.48%

Table VI. Comparison of hardware resource consumption and metrics of different Weak PUF designs.

PUF design	U´	R´	Hardware	Resp (bit)	Consumption
SRAM PUF [Guajardo et al. 2007]	49.97%	$> 88\%^t$	FPGA	128	4600 SRAM memory bits
Latch PUF [Su et al. 2008]	50.55%	96.96%	0.13um CMOS	128	1 latch for each ID cell
Latch PUF [Yamamoto et al. 2011]	46%	$> 87\%^t$	Spartan 3	128	2×128 slices
Flip-flop PUF [Maes et al. 2008]	$\approx 50\%^*$	$> 95\%^v$	Virtex 2	4096	4096 flip flops
Flip-flop PUF [van der Leest et al. 2010]	36%	$> 87\%^t$	ASIC	1024	1024 flip flops
Buskeeper PUF [Simons et al. 2012]	49%	$> 80\%^t$, $> 95\%^v$	TSMC 65 nm	192	$1GE$[1]
Butterfly PUF [Kumar et al. 2008]	$\approx 50\%^*$	94%	Virtex 5	64	130 slices
RO PUF [Suh and Devadas 2007]	46.15%	99.52%	Virtex 4	128	16×64 $array$[2]
CRO PUF [Merli et al. 2010]	43.50%	$> 96\%^t$, $\approx 100\%^*$	Spartan 3	127	64 slices for ROs except counters
PUF ID generator [Gu et al. 2014]	48.52%(S-6), 49.90%(A-7)	93.21%(S-6), 93.93%(A-7)	Spartan 6, Artix 7	128	128 slices
Ultra-compact PUF ID generator [Gu and O'Neill 2015]	49.93%	93.96%	Spartan 6	128	40 slices
Proposed improved PUF ID generator	45.60%(S-6)	99.42%(M), 98.74%(A)	Spartan 6, Artix 7	128	128 slices

[1]GE represented gate equivalent. [2]16×64 array = 1024 ROs; each RO consisting of 5 inverters and 1 AND. U´is uniqueness, R´ is reliability. Resp is response. ^{t} under temperature variation. ^{v} under supply voltage variation. * required post-processing. M is manual characterisation process, A is automated characterisation process.

Table V lists the results of all the evaluated PUF ID generator metrics. As previously mentioned, the reliability result has significantly improved using characterisation and a majority function, and is very close to the ideal value of 100%, with the uniformity also improving. The uniqueness decreases slightly using the improved PUF ID generator design; however this could be optimized by changing the position of the 1-bit ID cell

hard-macro to also balance the percentage of 1s and 0s in each response. Bit-aliasing does not occur in any of the designs.

6.6. Hardware Resources and Performance Analysis

For the manual characterisation process, ten identical Xilinx Spartan-6 boards were tested, each assembled with identical parts and components. There are a total of $1,430$ slices on a Spartan-6 LX9 FPGA, where each slice contains four LUTs and eight flip-flops. Each ID bit response generation design only needs one slice; hence, our 128-bit identification generator without error correction circuitry occupies only $\frac{128}{1430} \times 100\% = 8.95\%$ of the total slice resource, and $\frac{128+128}{1430} = 17.90\%$ with error correction circuitry. As can be seen the resource usage is minimal, even on a small FPGA. The layout is controlled using hard macros, which helps achieve the minimal resource footprint.

For the automated characterisation process, ten identical Xilinx Artix-7 boards were tested. Although a Xilinx Artix-7 FPGA does not have a SLICEX, its SLICEL and SLICEM are essentially the same as that of a Xilinx Spartan-6. The SLICEL and SLICEM of the Xilinx Artix-7 FPGA include all of the components of a SLICEX; hence the 1-bit PUF ID generator can be implemented on either the SLICEM or SLICEL also. In this experiment, a SLICEL is used to implement the hard macro. There are a total of $15,850$ slices on a Xilinx Artix-7 XC7A100T FPGA. Each 1-bit PUF ID generator design only needs one slice; hence, the 128-bit PUF ID generator design without majority voting circuitry occupies only $\frac{128}{15,850} \times 100\% = 0.81\%$ of the total slice resource available on this FPGA.

The resource usage comparison between the proposed PUF ID generator and other Weak PUFs implemented on hardware devices is shown in Table VI. The SRAM PUF proposed by Guajardo *et al.* [Guajardo et al. 2007], using SRAM memory cell, can return a response on power-up. The Latch PUF proposed by Su *et al.* [Su et al. 2008] is implemented on an ASIC not FPGA. The Flip-flop PUF proposed by Maes *et al.* [Maes et al. 2008], similar to SRAM, uses the power-up values of the flip-flops, however its randomness is limited and post-processing is required. The Butterfly PUF [Kumar et al. 2008], which is also suitable for FPGA implementation as it can be implemented using basic logic gates, reported 94% reliability over temperature variations. However reliability over voltage changes is not provided. It consumes 130 slices of a Xilinx Virtex-5 FPGA device for a 64-bit response generation, hence uses twice the hardware resources of the proposed 1-bit PUF ID generator design. The RO PUFs [Merli et al. 2010; Suh and Devadas 2007] and the CRO PUF [Merli et al. 2010] have been implemented on different FPGAs, e.g. Xilinx Virtex-4 and Spartan-3. The hardware resource consumption is at least 384 slices for a 64-bit response. It can be seen that the proposed PUF ID generator design is the most lightweight FPGA-based Weak PUF design reported to date. Moreover, the performance results for uniqueness and reliability show the effectiveness of the proposed PUF design.

7. CONCLUSIONS

In this paper, we have shown that an effective, reliable and low-cost PUF ID generator design is achievable for an FPGA device. A single ID cell fits efficiently within one FPGA slice and can be tailored for instantiation as a hard-macro to achieve balanced routing. The design is the most compact FPGA-based Weak PUF architecture reported to date. An example 128-bit PUF ID generator is implemented on both a Xilinx Spartan-6 and Artix-7 FPGA. It utilizes only half a slice for each 1-bit ID cell which consumes only 8.95% of the overall hardware resources of the Spartan-6 device and 0.81% of the Artix-7 device. The manual characterisation post-processing enhances the reliability of the 128-bit PUF ID generator design from 93.21% to 99.42%

without the requirement of any additional hardware resources. A further improvement to 99.78% is achieved when majority voting is also employed. An automated characterisation method is presented, which improves the reliability of the 128-bit PUF ID generator from 93.93% to 98.74% without the requirement of any additional hardware resources, and 99.60% for the design that employs a majority voting. Overall, experimental results demonstrate high uniqueness, reliability, uniformity and no bit-aliasing with values of 45.60%, 50.60% and 56.48% using characterisation process, and values of 45.60%, 50.54% and 56.58% using a further majority voting.

ACKNOWLEDGMENTS

This work has been supported by the KeyHAS project, the R&D program of IITP/MSIP (Study on secure key hiding technology for IoT devices), and by the SPARKS project, funded by EU 7th Framework Programme (FP7/2007-2013, grant agreement no. 608224; www.project-sparks.eu).

REFERENCES

Georg T. Becker. 2015. *The Gap Between Promise and Reality: On the Insecurity of XOR Arbiter PUFs*. Springer Berlin Heidelberg, Berlin, Heidelberg, 535–555. DOI: http://dx.doi.org/10.1007/978-3-662-48324-4_27

Mudit Bhargava and Ken Mai. 2014. An efficient reliable PUF-based cryptographic key generator in 65nm CMOS. In *Proceedings of the ACM/IEEE Design, Automation and Test in Europe Conference and Exhibition (DATE'14)*. 1–6. DOI: http://dx.doi.org/10.7873/DATE.2014.083

Christoph Bohm, Maximilian Hofer, and Wolfgang Pribyl. 2011. A microcontroller SRAM-PUF. In *Proceedings of the 5th International Conference on Network and System Security (NSS'11)*. 269–273.

Qingqing Chen, Gyorgy Csaba, Paolo Lugli, Ulf Schlichtmann, and Ulrich Ruhrmair. 2011. The Bistable Ring PUF: A new architecture for strong Physical Unclonable Functions. In *Proceedings of the IEEE International Symposium on Hardware-Oriented Security and Trust (HOST'11)*. San Diego, CA, 134–141. DOI: http://dx.doi.org/10.1109/HST.2011.5955011

Mafalda Cortez, Said Hamdioui, Vincent van der Leest, Roel Maes, and Geert-Jan Schrijen. 2013. Adapting voltage ramp-up time for temperature noise reduction on memory-based PUFs. In *Proceedings of the IEEE International Symposium on Hardware-Oriented Security and Trust (HOST'13)*. 35–40. DOI: http://dx.doi.org/10.1109/HST.2013.6581562

Jeroen Delvaux and Ingrid Verbauwhede. 2014. Fault Injection Modeling Attacks on 65 nm Arbiter and RO Sum PUFs via Environmental Changes. *IEEE Trans. Circuits Syst. I, Reg. Papers* 61, 6 (June 2014), 1701–1713. DOI: http://dx.doi.org/10.1109/TCSI.2013.2290845

Achiranshu Gary and Tony T. Kim. 2014. Design of SRAM PUF with improved uniformity and reliability utilizing device aging effect. In *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS'14)*. IEEE, Melbourne, Australia, 1941–1944.

Blaise Gassend, Dwaine Clarke, Marten van Dijk, and Srinivas Devadas. 2002. Silicon Physical Random Functions. In *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS '02)*. ACM, New York, NY, USA, 148–160. DOI: http://dx.doi.org/10.1145/586110.586132

Chongyan Gu, Yijun Cui, Neil Hanley, and Máire O'Neill. 2016. Novel Lightweight FF-APUF Design for FPGA. In *Proceedings of 29th IEEE International System-on-Chip Conference, (SOCC'16)*. Seattle, WA, USA.

Chongyan Gu, Julian Murphy, and Máire O'Neill. 2014. A unique and robust single slice FPGA identification generator. In *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS'14)*. Melbourne, Australia, 1223–1226. DOI: http://dx.doi.org/10.1109/ISCAS.2014.6865362

Chongyan Gu and Máire O'Neill. 2015. Ultra-Compact and Robust FPGA-Based PUF Identification Generator. In *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS'15)*. Lisbon, Portugal. DOI: http://dx.doi.org/10.1109/ISCAS.2015.7168788

Jorge Guajardo, Sandeep S. Kumar, Geert-Jan Schrijen, and Pim Tuyls. 2007. *FPGA Intrinsic PUFs and Their Use for IP Protection*. Springer Berlin Heidelberg, Berlin, Heidelberg. 63–80 pages. DOI: http://dx.doi.org/10.1007/978-3-540-74735-2_5

Charles Herder, Meng-Day Yu, Farinaz Koushanfar, and Srinivas Devadas. 2014. Physical unclonable functions and applications: A tutorial. *Proc. IEEE* 102, 8 (2014), 1126–1141.

Daniel E Holcomb, Wayne P Burleson, and Kevin Fu. 2009. Power-up SRAM state as an identifying fingerprint and source of true random numbers. *IEEE Trans. Comput* 58, 9 (2009), 1198–1210.

Daniel E. Holcomb, Amir Rahmati, Mastooreh Salajegheh, Wayne P. Burleson, and Kevin Fu. 2013. *DRV-Fingerprinting: Using Data Retention Voltage of SRAM Cells for Chip Identification*. Springer Berlin Heidelberg, Berlin, Heidelberg, 165–179. DOI: http://dx.doi.org/10.1007/978-3-642-36140-1_12

Yohei Hori, Hyunho Kang, Toshihiro Katashita, Akashi Satoh, Shinichi Kawamura, and Kazukuni Kobara. 2014. Evaluation of Physical Unclonable Functions for 28-nm Process Field-Programmable Gate Arrays. *Journal of Information Processing* 22, 2 (2014), 344–356.

Intrisic-ID. accessed 17th June 2015. NXP Secures Over Two Billion Payment and Government ID Cards with SmartMX. (accessed 17th June 2015). https://www.intrinsic-id.com/nxp-secures-over-two-billion-payment-and-government-id-cards-/with-smartmx/

Raghavan Kumar and Wayne Burleson. 2014. On design of a highly secure PUF based on non-linear current mirrors. In *Proceedings of the IEEE International Symposium on Hardware-Oriented Security and Trust HOST'14, Arlington, VA, USA, May 6-7, 2014*. IEEE Computer Society, 38–43. DOI: http://dx.doi.org/10.1109/HST.2014.6855565

Sandeep S Kumar, Jorge Guajardo, Roel Maes, G-J Schrijen, and Pim Tuyls. 2008. The butterfly PUF protecting IP on every FPGA. In *Proceedings of the IEEE International Symposium on Hardware-Oriented Security and Trust (HOST'08)*. 67–70. DOI: http://dx.doi.org/10.1109/HST.2008.4559053

Klaus Kursawe, Ahmad-Reza Sadeghi, Dries Schellekens, Boris Skoric, and Pim Tuyls. 2009. Reconfigurable Physical Unclonable Functions - Enabling Technology for Tamper-resistant Storage. In *Proceedings of the IEEE International Workshop on Hardware-Oriented Security and Trust (HST '09)*. IEEE Computer Society, Washington, DC, USA, 22–29. DOI: http://dx.doi.org/10.1109/HST.2009.5225058

Daihyun Lim, Jae W Lee, Blaise Gassend, G Edward Suh, Marten Van Dijk, and Srinivas Devadas. 2005. Extracting secret keys from integrated circuits. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst* 13, 10 (2005), 1200–1205.

Roel Maes, Pim Tuyls, and Ingrid Verbauwhede. 2008. Intrinsic PUFs from Flip-flops on Reconfigurable Devices. In *Proceedings of the 3rd Benelux Workshop on Information and System Security (WISSec'08)*. Eindhoven,NL, 17.

Ahmed Mahmoud, Ulrich Rührmair, Mehrdad Majzoobi, and Farinaz Koushanfar. 2013. Combined Modeling and Side Channel Attacks on Strong PUFs. *IACR Cryptology ePrint Archive* 2013 (2013), 632. http://eprint.iacr.org/2013/632

Abhranil Maiti, Inyoung Kim, and Patrick Schaumont. 2012. A robust physical unclonable function with enhanced challenge-response set. *IEEE Trans. Inf. Forensics Security* 7 (2012), 333–345.

Abhranil Maiti and Patrick Schaumont. 2012. A novel microprocessor-intrinsic Physical Unclonable Function. In *Proceedings of the 22nd International Conference on Field Programmable Logic and Applications (FPL'12)*. Oslo, Norway, 380–387. DOI: http://dx.doi.org/10.1109/FPL.2012.6339208

Mehrdad Majzoobi, Akshat Kharaya, Farinaz Koushanfar, and Srinivas Devadas. 2014. Automated Design, Implementation, and Evaluation of Arbiter-based PUF on FPGA using Programmable Delay Lines. *IACR Cryptology ePrint Archive* 2014 (2014), 639. http://eprint.iacr.org/2014/639

Dominik Merli, Frederic Stumpf, and Claudia Eckert. 2010. Improving the Quality of Ring Oscillator PUFs on FPGAs. In *Proceedings of the 5th Workshop on Embedded Systems Security (WESS '10)*. ACM, New York, NY, USA, Article 9, 9 pages. DOI: http://dx.doi.org/10.1145/1873548.1873557

Microsemi. accessed 17th June 2015. Microsemi SmartFusion2 SoC FPGAs Offer More Resources in Low Density Devices With The Lowest Power, Proven Security and Exceptional Reliability. (accessed 17th June 2015). http://www.microsemi.com/products/fpga-soc/soc-fpga/smartfusion2

Julian Murphy, Máire O'Neill, Frank Burns, Alex Bystrov, Alexandre Yakovlev, and Basel Halak. 2012. Self-Timed Physically Unclonable Functions. In *Proceedings of the 5th IFIP International Conference on New Technologies, Mobility and Security (NTMS'12)*. Istanbul, Turkey, 1–5.

Ravikanth Pappu, Ben Recht, Jason Taylor, and Neil Gershenfeld. 2002. Physical one-way functions. *Science* 297, 5589 (2002), 2026–2030.

Ed Peterson. 2015. Leveraging Asymmetric Authentication to Enhance Security-Critical Applications Using Zynq-7000 All Programmable SoCs. (Oct. 2015). http://www.xilinx.com/support/documentation/white_papers/wp468_asym-auth-zynq-7000.pdf

Ulrich Rührmair, Frank Sehnke, Jan Sölter, Gideon Dror, Srinivas Devadas, and Jürgen Schmidhuber. 2010. Modeling Attacks on Physical Unclonable Functions. In *Proceedings of the 17th ACM Conference on Computer and Communications Security (CCS '10)*. ACM, New York, NY, USA, 237–249. DOI: http://dx.doi.org/10.1145/1866307.1866335

Peter Simons, Erik van der Sluis, and Vincent van der Leest. 2012. Buskeeper PUFs, a promising alternative to D flip-flop PUFs. In *Proceedings of the IEEE International Symposium on Hardware-Oriented Security and Trust (HOST'12)*. 7–12. DOI: http://dx.doi.org/10.1109/HST.2012.6224311

Nicolas Sklavos. 2013. *Securing Communication Devices via Physical Unclonable Functions (PUFs)*. Springer Fachmedien Wiesbaden, Wiesbaden, 253–261. DOI: http://dx.doi.org/10.1007/978-3-658-03371-2_22

Ying Su, Jeremy Holleman, and Brian P Otis. 2008. A digital 1.6 pJ/bit chip identification circuit using process variations. *IEEE J. Solid-State Circuits* 43 (2008), 69–77.

G. Edward Suh and Srinivas Devadas. 2007. Physical Unclonable Functions for Device Authentication and Secret Key Generation. In *Proceedings of the 44th Annual Design Automation Conference (DAC '07)*. ACM, New York, NY, USA, 9–14. DOI: http://dx.doi.org/10.1145/1278480.1278484

Vincent van der Leest, Geert-Jan Schrijen, Helena Handschuh, and Pim Tuyls. 2010. Hardware Intrinsic Security from D Flip-flops. In *Proceedings of the 5th ACM Workshop on Scalable Trusted Computing (STC '10)*. ACM, New York, NY, USA, 53–62. DOI: http://dx.doi.org/10.1145/1867635.1867644

Arunkumar Vijayakumar and Sandip Kundu. 2015. A novel modeling attack resistant PUF design based on non-linear voltage transfer characteristics. In *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition, DATE 2015, Grenoble, France, March 9-13, 2015*, Wolfgang Nebel and David Atienza (Eds.). ACM, 653–658. http://dl.acm.org/citation.cfm?id=2755903

David Wolpert and Paul Ampadu. 2012. Temperature effects in semiconductors. In *Managing Temperature Effects in Nanoscale Adaptive Systems*. Springer, 15–33.

Xilinx. 2011. Spartan-6 Family Overview. (2011). http://www.xilinx.com/support/documentation/data_sheets/ds160.pdf Accessed 29 April 2016.

Dai Yamamoto, Kazuo Sakiyama, Mitsugu Iwamoto, Kazuo Ohta, Takao Ochiai, Masahiko Takenaka, and Kouichi Itoh. 2011. Uniqueness Enhancement of PUF Responses Based on the Locations of Random Outputting RS Latches. In *Cryptographic Hardware and Embedded Systems – CHES 2011: 13th International Workshop, Nara, Japan, September 28 – October 1, 2011. Proceedings*, Bart Preneel and Tsuyoshi Takagi (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 390–406. DOI: http://dx.doi.org/10.1007/978-3-642-23951-9_26

Haile Yu, Philip H. W. Leong, and Qiang Xu. 2012. An FPGA chip identification generator using configurable ring oscillators. *IEEE Trans. Very Large Scale Integr. Syst.* 20, 12 (Dec. 2012), 2198–2207. DOI: http://dx.doi.org/10.1109/TVLSI.2011.2173770